LA-UR -91-729

CONF-910295--1

LA-UR--91-729

DE91 008608

TITLE Application of Neural Networks and Information Theory to the Identification of *E. coli* Transcriptional Promoters

AUTHOR(S) Kenneth Abremski, DuPont Merck Pharmaceutical Company Experimental Station, Wilmington, DE 19880-0328

Karl Sirotkin, National Center for Biotechnology Information, National Institutes of Health, Bethesda, MD 20894

Alan Lapedes, Theoretical Division, Los Alamos National Laboratory Los Alamos, NM 87545

SUBMITTED TO Eighth International Conference on Mathematical and Computer Modelling Held February 14, 1991

Los Alamos Los Alamos National Laboratory Los Alamos, New Mexico 87545

FORM NO 836 R4
ST NO 2629 5/81

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Application of Neural Networks and Information Theory to the Identification of *E. coli* Transcriptional Promoters

Kenneth Abremski
Du Pont Merck Pharmaceutical Company
Experimental Station
P.O. Box 80328
Wilmington, DE 19880-0328

Karl Sirotkin
National Center for Biotechnology Information
National Institutes of Health
Bethesda, MD 20894

Alan Lapedes
Theoretical Division
Los Alamos National Laboratory
MS B213
Los Alamos, NM 87545

Submitted to: Eighth International Conference on Mathematical and Computer Modelling

Date: Thu, Feb 14, 1991

Application of Neural Networks and Information Theory to the Identification of *E. coli* Transcriptional Promoters

Kenneth Abremski
Du Pont Merck Pharmaceutical Company
Experimental Station
P.O. Box 80328
Wilmington, DE 19880-0328

Karl Sirotkin
National Center for Biotechnology Information
National Institutes of Health
Bethesda, MD 20894

Alan Lapedes
Theoretical Division
Los Alamos National Laboratory
MS B213
Los Alamos, NM 87545

Abstract

The Human Genome Project has as its eventual goal the determination of the entire DNA sequence of man, which comprises approximately 3 billion base pairs. An important aspect of this project will be the analysis of the sequence to locate regions of biological importance. New computer methods will be needed to automate and facilitate this task. In this paper, we have investigated the use of neural networks for the recognition of functional patterns in biological sequences. The prediction of *Escherichia coli* transcriptional promoters was chosen as a model system for these studies. Two approaches were employed. In the first method, a mutual information analysis of promoter and non-promoter sequences was carried out to determine the informative base positions that help to distinguish promoter sequences from non-promoter sequences. These base positions were then used to train a Perceptron to predict new promoter sequences. In the second method, the experimental knowledge of promoters was used to indicate the important base positions in the sequence. These base positions were used to train a back propagation network with hidden units which represented regions of sequence conservation found in promoters. With both types of networks, prediction of new promoter sequences was greater than 96.9%

Introduction

The increase in the number of biological sequences necessitates the development of computer methods for the prediction and analyses of important functional regions. These analyses should identify significant patterns to guide researchers in their experimental efforts to elucidate the functional role of a particular nucleic acid or protein sequence. There are a number of computer methods that have been applied to the recognition of specific patterns in biological sequences. These include consensus sequence matching, probability matrices, and various scoring techniques based on nucleic acid or protein similarities (for reviews see Waterman, 1989; Doolittle, 1990).

In this study we have investigated the use of various neural network architectures for the recognition of transcriptional promoter sequences derived from the bacterium *Escherichia coli*. *E. coli* promoters can be recognized by two somewhat conserved six-base sequences, termed the -35 and -10 regions (see Figure 1A). The numbering refers to their approximate distances from the start of transcription. The -35 region has the consensus sequence "ttgaca", and the -10 region has the consensus sequence "tataat". Most promoters do not contain these exact sequences. The separation distance between these two regions can vary from 15 to 21 bases, with an average of 17-18 bases. Previous analyses of promoter sequences used consensus sequence matching and base probability matrices to predict promoter sequences (Hawley and McClure,1983; Mulligan, et al., 1984; Harley and Reynolds, 1987; Rozkot, et al., 1989; O'Neill and Chiafari, 1989). Several studies have used a neural network algorithm and used only the -10 and -35 regions to train the networks (Nakata et al., 1988; Lukashin, et al., 1989; Alexandrov and Mironov, 1990).

We have combined a neural network approach with two different techniques for selecting the data that is used to train the neural net. In the first case, we have carried out a statistical analysis of promoter and non-promoter sequences. From these results, we have used the most informative base positions to train the neural network. In the second case, we have used our biological knowledge of promoters to set up a neural network architecture.

System and methods

Programs

Programs were written in the C programming language. Programming was carried out on a Sun SPARCstation 1 computer from SUN Microsystems, Inc. with the Unix operating system, SunOS 4.0.

Sequence Information

Candidate promoter sequences were obtained from the compilation of Harley and Reynolds (1987). We have removed sequences that relate to mutant promoters and to promoters that use different sigma factors (e.g. the heat shock promoters). Sequence information for these promoters was obtained from GenBank, release 62.0. Each of the 128 promoters included 90 bases of sequence around the conserved -10 and -35 regions.

For the training of the neural network it was necessary to generate a set of sequences which were not promoters. To do this we selected at random 90 base sequences from various coding regions of known *E. coli* genes. These coding sequences were used as examples of non-promoters. They included sequences from the following genes: *lac*ZYA, *ala*S, *trp*S, *avt*A, *ace*EF, *lpd*, *frd*AB, *amp*C, *asp*A, *pfk*A, *sbp*, *cdh*, *cya*A, *dna*AN, *phe*A, *tyr*A, *aro*F, *mal*EFK, *lam*B, and *his*GCDHAFIE.

Mutual Information

The sequences were aligned by their -10 regions as published in Harley and Reynolds (1987). The set of promoters contained 128 sequences and the set of non-promoters contained 1300 sequences. From these two classes of sequences two data sets were constructed. Each set contained 64 promoters and various numbers of non-promoters. The number of non-promoters was varied between 64 and 625. There were no sequences in common between the two data sets. One of these, called the training set, was used to train the neural network and the second, called the testing set, was used to test it. The single letter designation of bases, "a, c, g, t" was converted to a "unary" notation, where "a" is replaced with the binary string "0001", "c" with "0010", "g" with "0100" and "t" with "1000". Thus each base position has a 4 bit unary code denoting the presence or absence of the bases "a, c, g, t". Since only one base appears at each position, each group of four bits has only one "1" indicating the presence of one of the bases "a, c, g, t", depending on whether the first, second, third, or fourth bit is set to "1". The mutual information of each of these bit positions within the class, "T" (for promoters) and "F" (for non-promoters), was calculated for the training set according to equation (1).

$$I = \sum_{C=T,F} \sum_{b=0,1} \text{Prob}[C,b] \times \log_2 \frac{\text{Prob}[C,b]}{\sum_{C=T,F}\text{Prob}[C,b] \times \sum_{b=0,1}\text{Prob}[C,b]} \qquad (1)$$

C is the class of sequence, "T" for promoter and "F" for non-promoter, and "b = 0,1" denotes the presence or absence of bases "a, c, g, t" at each base position. Equation (1) measures the mutual information between bit positions and the class, and can be interpreted as measuring the importance of the presence, or absence, of each base in each position for determining class. We have also measured the mutual information for pairs of bits where the same formula applies but the "b" summation is over the four possibilities "00, 01, 10, 11". This quantity measures the importance in determining class of the presence or absence of pairs of bases in the various positions, and is of interest because of the possible interaction of the -10 and -35 regions in determining whether or not a sequence contains a promoter. The real valued mutual information scores may be ranked from high to low values, with high values identifying a base position, and also the base in that position, whose presence or absence is informative about the class. When the mutual information is applied to pairs of bits one may determine the important pairs of base positions, and the bases in those positions, whose presence or absence is important to determining class.

When sequences were later input to a Perceptron network only those bit positions of the unary coded data were used that have mutual information above a certain arbitrary cutoff value. For example, one may choose a cutoff so that only the top 10 bit positions are used by the network. This has the effect of reducing the number of weights in the network, which is often an aid to achieving good generalization outside the training set, and of course, the actual bit positions determined by the mutual information may be examined in the context of biological knowledge. The highest fifty mutual information values of bit, and bit pairs, are presented in Table 1. The -10 region corresponds to positions 53-58 in Table 1, while the variable -35 region corresponds to positions 26-38. (See Figure 1A). Not unexpectedly, the highest values include mainly positions in the -10 and -35 consensus regions. The input to the Perceptron network is therefore a sequence of 0's and 1's, obtained be choosing only those bit positions over a certain cutoff.

Perceptron

In the Perceptron neural network, there are two layers of processing units, an input layer representing the sequences and an output layer which signals the class that the sequence belongs to, either a promoter or non-promoter (see Figure 1B). A Perceptron is trained using the sequence information translated into numerical values based on the mutual information analyses as described above. This type of network was previously used for recognition of ribosome binding sites (Stormo et al., 1982). In the Perceptron, the output layer consists of a single unit. Its value can vary between 0 and 1 and is calculated according to equations (2a) and (2b) (Rumelhart et al., 1986).

$$\text{net}_p = \sum_{i=1}^{N} \text{weight}_i \times \text{input}_i + \text{bias} \qquad (2a)$$

$$\text{output}_p = \frac{1}{1 + e^{-\text{net}_p/T}} \qquad (2b)$$

The bias term is equivalent to a learned weight that is connected to an input unit that always has a value of 1. The above equation is somewhat different from the original Perceptron formalism in that a sigmoidal threshold function was used to generate the output value, rather than a linear step function. A value >= 0.5 predicts a promoter, while a value < 0.5 predicts a non-promoter sequence. In all of the tables the results are presented as the percent correctly predicted. This is calculated from the number of promoters with an output greater than or equal to 0.5, divided by the total number of promoters. Similar calculations are made for the non-promoter sequences. The number of input units was varied between 10 and 50 depending on the number of single and pairwise bit positions that was used. A series of real valued weights connects each input unit to the output unit. During training, the error was calculated over all the patterns as shown in equation (3), as the difference between the expected target value of the output unit (0.9 in the case of a promoter and 0.1 in the case of a non-promoter) and the calculated output.

$$\text{Error} = \frac{1}{2} \sum_{p=1}^{N} (\text{target}_p - \text{output}_p)^2 \qquad (3)$$

If the error was greater than some level (usually 10 percent of the initial error at the start of training), then the values of the weights were changed to minimize the error by using a gradient descent method (Rumelhart et al., 1986). The weights were changed according to the learning rule given in equation (4).

$$\Delta W_i = -k \frac{\partial E_p}{\partial w_i} \qquad (4)$$

Table 1. 50 Highest Positions of Mutual Information

Rank	NumBits	Positions	Bases
1	2	58 : 53	t,Not_t : t, Not_t
2	2	54 : 53	a,Not_a : t,Not_t
3	2	58 : 54	t,Not_t : a,Not_a
4	1	53	t,Not_t
5	1	58	t,Not_t
6	2	56 : 53	a,Not_a : t,Not_t
7	1	54	a,Not_a
8	2	58 : 56	t,Not_t : a,Not_a
9	2	53 : 31	t,Not_t : t,Not_t
10	2	56 : 54	a,Not_a : a,Not_a
11	2	54 : 30	a Not_a : t,Not_t
12	2	57 : 53	a,Not_a : t,Not_t
13	2	55 : 53	t,Not_t : t,Not_t
14	2	58 : 22	t,Not_t : a,Not_a
15	2	58 : 30	t,Not_t : t,Not_t
16	2	58 : 31	t,Not_t : t,Not_t
17	2	53 : 30	t,Not_t : t,Not_t
18	2	53 : 22	t,Not_t : a,Not_a
19	2	54 : 22	a,Not_a : a,Not_a
20	2	53 : 6	t,Not_t : t,Not_t
21	2	54 : 31	a,Not_a : t,Not_t
22	2	53 : 15	t,Not_t : t,Not_t
23	2	54 : 50	a,Not_a : t,Not_t
24	2	81 : 53	a,Not_a : t,Not_t
25	2	85 : 53	a,Not_a : t,Not_t
26	2	85 : 54	a,Not_a : a,Not_a
27	2	88 : 53	g,Not_g : t,Not_t
28	2	58 : 55	t,Not_t : t,Not_t
29	2	53 : 13	t,Not_t : t,Not_t
30	2	58 : 26	t,Not_t : t,Not_t
31	2	53 : 26	t,Not_t : t,Not_t
32	2	53 : 48	t,Not_t : t,Not_t
33	2	58 : 15	t,Not_t : t,Not_t
34	2	56 : 31	a,Not_a : t,Not_t
35	2	76 : 53	a,Not_a : t,Not_t
36	2	67 : 58	a,Not_a : t,Not_t
37	2	53 : 16	t,Not_t : a,Not_a
38	2	58 : 16	t,Not_t : a,Not_a
39	2	55 : 54	t,Not_t : a,Not_a
40	2	58 : 50	t,Not_t : t,Not_t
41	2	53 : 51	t,Not_t : g,Not_g
42	2	53 : 5	t,Not_t : t,Not_t
43	2	53 : 29	t,Not_t : t,Not_t
44	2	53 : 11	t,Not_t : t,Not_t
45	1	56	a,Not_a
46	2	81 : 54	a,Not_a : a,Not_a
47	2	53 : 50	t,Not_t : t,Not_t
48	2	58 : 11	t,Not_t : t,Not_t
49	1	58	g,Not_g
50	1	53	c,Not_c

Expansion of this equation in terms of the output and weight values gives equations (5a) and (5b). This shows how the changes in the values of the weights depend on the values of the output, the expected target, and the previous weight. In equation (5a), ε is the value for the learning rate and α is the value for the momentum term used during training. These values were usually set to 0.05 and 0.90, respectively. These constants affect the rate of learning during the training procedure.

$$\Delta W_i(n+1) = \varepsilon \times \delta_i \times \text{output}_i + \alpha \times \Delta W_i(n) \tag{5a}$$

$$\delta_i = (\text{target}_i - \text{output}_i) \times \text{output}_i \times (1 - \text{output}_i) \tag{5b}$$

Experimental information

In the second method, we know from experimental work that the -10 and -35 regions are important for promoter recognition (Hawley and McCLure, 1983). Therefore, only bases surrounding these regions are used as input for the neural network. For the -10 region we use 6 bases at the positions that coincide with a -10 consensus region. From the analysis of many promoter sequences, it is known that the distance between the -35 region and the -10 region is variable. This spacing is usually 17-18 bases but can vary between 15 and 21 bases. Therefore, we have used 12 bases beginning 27 bases away from the -10 region. This set of bases should contain information for all possible -35 regions beginning 15 to 21 bases away from the -10 region. As input for the network, these 18 bases are extracted from the promoter sequences and are converted according to the following coding scheme "a" = "0001", "c" = "0010", "g" = "0100", and "t" = "1000". This results in an array of 0's and 1's for each sequence, of length 72. The remainder of the bases are not included. This data was used to train the back propagation network which is described below.

Back Propagation

The back propagation network consisted of 3 layers of units, the input layer containing 72 units, a hidden layer of 8 units and an output layer of a single unit (see Figure 1C). The 8 hidden units were chosen to represent the 7 possible -35 regions, i.e. those separated by 15-21 bases, and the -10 region. Not all input units were connected to every hidden unit. The input units 1-24 were connected to the first hidden unit, input units 5-28 were connected to the second hidden unit, input units 9-32 were connected to the third hidden unit, etc. Finally, the last 24 input units were connected to the eighth hidden unit. The 24 weights from the input units to each of the first seven hidden units comprise feedforward subnets in the full network architecture. Each of these seven subnets was constrained to have the same weights so that the hidden units can respond to the presence or absence of a -35 signal in a translationally invariant manner. Thus, the weights in these subnets were initialized identically, and constrained during training to have the identical weight values. Therefore, there is really only one distinct subnet for these connections, and this single subnet is duplicated over the connections of the first seven hidden units to the inputs. The total number of weights is 56, 48 between the input and hidden layers and 8 between the hidden and output layers. There were also three bias terms. The training was carried out using similar error minimization and weight change rules to those described above (Rumelhart et al., 1986).

Interactions between -10 and -35 regions

To investigate whether the -10 and -35 regions show any interaction we have trained a Perceptron using these regions either alone or in combination. The network was trained on the -10 region alone or, on the -35 region alone, or on both regions together. For this procedure, the 6 bases of the -10 region and the 6 bases of the -35 region were used and converted with the 4 bit coding described above. This gave 24 input units when either the -10 or -35 regions were trained alone, or 48 input units when both were used.

Results

Mutual Information and Perceptron

For training the Perceptron, three cases were chosen using either the top 10, 30 or 50 values of mutual information. In the case where only ten values were used, this corresponded to 5 base positions within the 90 base promoter region. Using the top 30 values involved 17 positions and the top 50 values involved 25 positions. Table 2A shows the results after training the Perceptron. In all cases, the network predicts the training set with an accuracy greater than 96.9%. However, learning of the training set is better when 30 or 50 values of mutual information are used (98.4% versus 96.9%). As a measure of how well the network can generalize, the prediction of the network was determined using the testing data set. The results, shown in Table 2B, demonstrates that the network can predict new promoter sequences with an accuracy of 96.9%.

Table 2. Perceptron using Mutual Information

(A) Training Set

Input Units	Total	Promoter	Non-Promoter
10	96.1	96.9	95.3
30	99.2	98.4	100.0
50	99.2	98.4	100.0

(B) Testing Set

Input Units	Total	Promoter	Non-Promoter
10	97.7	96.9	98.4
30	98.4	96.9	100.0
50	97.7	96.9	98.4

The training and testing sets included 64 promoter and 64 non-promoter sequences There was no duplication of sequences between the two data sets. The numbers in columns 2-4 refer to the percent predicted correctly.

Table 3. Back Propagation Results

Data Set	Total	Promoter	Non-Promoter
Training	99.2	100.0	98.4
Testing	97.7	96.9	98.4

The training and testing sets included 64 promoter and 64 non-promoter sequences. The numbers in the first row correspond to the percent predicted correctly for the training data, while the second row corresponds to the percent correct for the testing data.

Back Propagation

The results using our experimental knowledge of promoters and the back propagation network are shown in Table 3. In this case there were 72 input units derived from the six bases comprising the -10 region and the 12 bases that comprise the 7 possible -35 regions. After training this network, promoters in the training set are predicted with 100% accuracy. On the testing set, the network can predict new promoters with an accuracy of 96.9%, which is equivalent to the Perceptron network described in the previous section.

Interactions between the -10 and -35 regions

Since the -10 and -35 regions of promoters are known to be important for function, and since these regions were used in previous studies (Lukashin et al., 1989), a Perceptron was trained using only the bases from these regions. Six bases from the -35 region and 6 bases from the -10 region were used. The network consisted of 48 input units and 1 output unit. Table 4A shows the results following training with a data set containing 64 promoters and 625 non-promoters. The overall prediction rate was 99.3% (-10 and -35). Similar results are obtained with the test set data (Table 4B), where the prediction rate was 99.5% (-10 and -35).

To determine if there might be some interaction between the -10 and -35 regions, we compared the test results from a network that was trained using both the -10 and -35 regions (-10 and -35) versus a network that used the weights derived from training with each region alone. To examine the importance of both conserved regions in determining a promoter sequence, we also trained a Perceptron using just the -10 region alone or just the -35 region. In this case the appropriate six bases were used and the Perceptron consisted of 24 input units and one output unit. The results from these networks (see Table 4A) show that the overall prediction was 95.6% on the training set and 95.7% on the testing set using the -10 region alone. For the -35 region, the results were not as good, with 89.4% overall prediction for training and 88.7% overall prediction for testing.

We tested whether the neural network could pick up any extra information during training by using both the -10 and -35 regions. To do this, we compared the results, using the testing data set, on the network derived from training with both regions versus a

Table 4. Perceptron -10 and -35 Region Interaction

(A) Training

Region	Total	Promoter	Non-Promoter
-10 Region	95.6	96.7	95.5
-35 Region	89.4	92.2	89.1
-10 and -35	99.3	98.4	99 4

(B) Testing

Region	Total	Promoter	Non-Promoter
-10 Region	95.7	92.2	96.1
-35 Region	88.7	93.7	88.2
-10 and -35	99.5	98.4	99.7
Combine -10, -35	98.5	85.9	99.8

The numbers correspond to the percent predicted correctly. For this table, only the -10 and -35 regions from the 64 promoters and corresponding regions from the 625 non-promoters were used for training and testing. The region refers to the region used for input to the networks. -10 refers to training in the presence of the -10 region alone. -35 refers to training in the presence of the -35 region alone. -10 and -35 refers to training in the presence of both the -10 and -35 consensus sequences. Combine -10, -35 refers to using the networks trained in the presence of the -10 alone and the -35 alone and combining them, then testing using this combined network.

network which combined the connection weights from the networks trained on the -10 region alone and trained on the -35 region alone. This network had 48 input units, 24 weights derived from the -10 region network, 24 weights derived from the -35 region, 2 bias terms, and 2 output units. If both output units gave values greater than or equal to 0.5, then the test sequence was classified as a promoter. The results are shown in Table 4B and indicate that promoter sequences are predicted with far less accuracy, 85.9% versus 98.4%, using the weights derived from networks that were trained on individual regions, then when a network was trained using both regions.

Discussion

This study investigated the utility of neural networks for learning to distinguish prokaryotic promoter sequences from non-promoter sequences.Two approaches were employed. In the first, the sequences of promoters and non-promoters in the training set were analyzed for the mutual information at each base position. This information indicated the base positions that best distinguished promoter sequences from non-promoter sequences. This information was then used to train a Perceptron. In the second case, the available experimental information on promoters was used to set up and train a back propagation network. With either type of network, the results were fairly similar and the trained networks predicted promoter and non-promoter sequences in a training and testing data set with an accuracy greater than 96%. For the prediction of promoters these results using neural networks seemed much higher than the results from previous studies which used statistical methods different from neural networks. Although the exact promoters used in these studies were not identical, there was a lot of overlap in the sequences used. The algorithm of Mulligan et al. (1984) predicted promoter sequences at the 83% level, while that of O'Neill and Chiafari (1989) correctly identified 77% of the promoters tested.

We also investigated the effects of training a neural network on each of the -10 and -35 consensus regions alone, versus using both regions. If there are no interactions, the weights from a network trained on the -10 region could be combined with the weights from a network trained on the -35 region to predict promoters. This level of prediction should be equivalent to a network that was trained using both regions. However, our results indicate that the network can learn to predict promoter sequences more effectively when both regions are present during the training.

A major part of the human genome effort will be the analyses of sequences to determine their biological functions. Efficient computer methods for pattern matching will need to be developed to carry out this task. From the results presented here, it appears that neural networks may prove useful for this problem and the method is general enough to be applicable to most patterns where there are a representative number of known examples.

References

Alexandrov, N.N. and A.A.Mironov. 1990. Application of a new method of pattern recognition in DNA sequence analysis: a study of *E. coli* promoters. Nucl. Acid. Res. **18**:1847-1852.

Doolittle, R.F. (ed.) 1990. Methods in Enzymology, Molecular Evolution: Computer Analysis of Protein and Nucleic Acid Sequences. **183**. Academic Press, Inc. San Diego, CA.

Harley, C.B. and R.P. Reynolds. 1987. Analysis of *E.coli* promoter sequences. Nucl. Acid. Res. **15**:2343-2361.

Hawley, D.K. and W.R. McClure. 1983. Compilation and analysis of *Escherichia coli* promoter DNA sequences. Nucl. Acid. Res. **11**:2237-2255.

Lukashin, A.V., Anshelevich, V.V., Amirikyan, B.R., Gragerov, A.I. and Frank-Kamenetskii, M.D. 1989. Neural Network Models for Promoter Recognition. Journal of Biomolecular Struct. & Dynam. **6**:1123-1133.

Mulligan, M.E., Hawley, D.K., Entriken, R. and W.R. McClure. 1984. *Escherichia coli* promoter sequences predict in vitro RNA polymerase selectivity. Nucl. Acid.Res. **12**:789-800.

Nakata, K., Kanehisa, M. and J.V.Maizel, Jr. 1988. Discriminant analysis of promoter regions in *Escherichia coli* sequences. CABIOS, **4**:367-371.

O'Neill, M.C. and F. Chiafari. 1989. *Escherichia coli* Promoters II. A Spacing Class-dependent Promoter Search Protocol. J. Biol. Chem. **264**:5531-5534.

Rozkot, F., Sazelova, P. and L. Pivec. 1989. A novel method for promoter search enhanced by function-specific subgrouping of promoters - developed and tested on *E.coli* system. Nucl. Acid. Res. **17**:4799-4815.

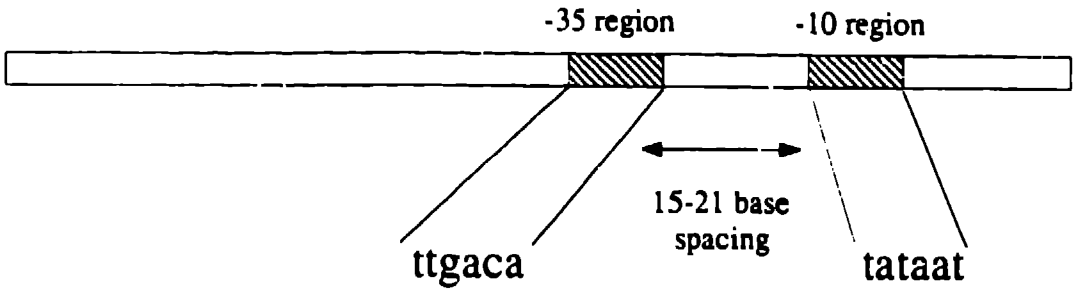
Rumelhart, D.E., McClelland, J.L. and the PDP Research Group. 1986. Parallel Distributed Processing, Explorations in the Microstructure of Cognition. Vol. 1: Foundations. MIT Press, Cambridge, Mass.

Stormo, G.D., Schneider, T.D., Gold, L. and A. Ehrenfeucht. 1982. Use of the Perceptron algorithm to distinguish translational initiation sites in *E. coli*. Nucl. Acid. Res. **10**:2997-3011.

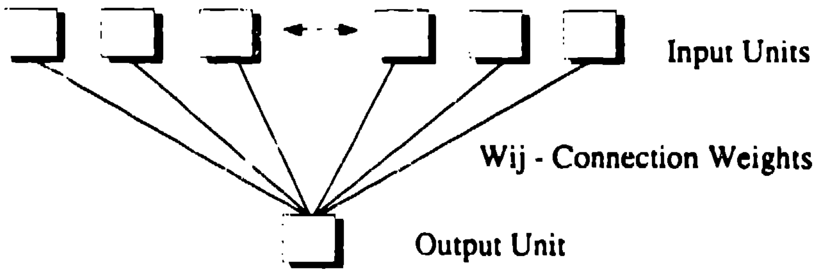
Waterman, M.S., 1989. Mathematical Methods for DNA Sequences. CRC Press, Boca Raton, Fla.

FIGURE 1. (A) E. coli Promoters. This is a diagram of the relevant features found in prokaryotic transcriptional promoters.The two conserved regions are indicated, and their consensus sequences are shown. -10 and -35 refers to their approximate location from the start of RNA transcription. These two conserved regions can be separated by a variable distance of between 15 and 21 bases. (B) Perceptron. This shows a Perceptron neural network architecture, consisting of input units, which are connected to a single output unit. The connection weights are represented by lines connecting an input unit to the output unit. (C) Back Propagation. The back propagation network contains three layers of processing units. The input units connect to units in the middle or hidden layer, and the hidden units connect to the output unit. In this network, not every input unit is connected to every hidden unit.

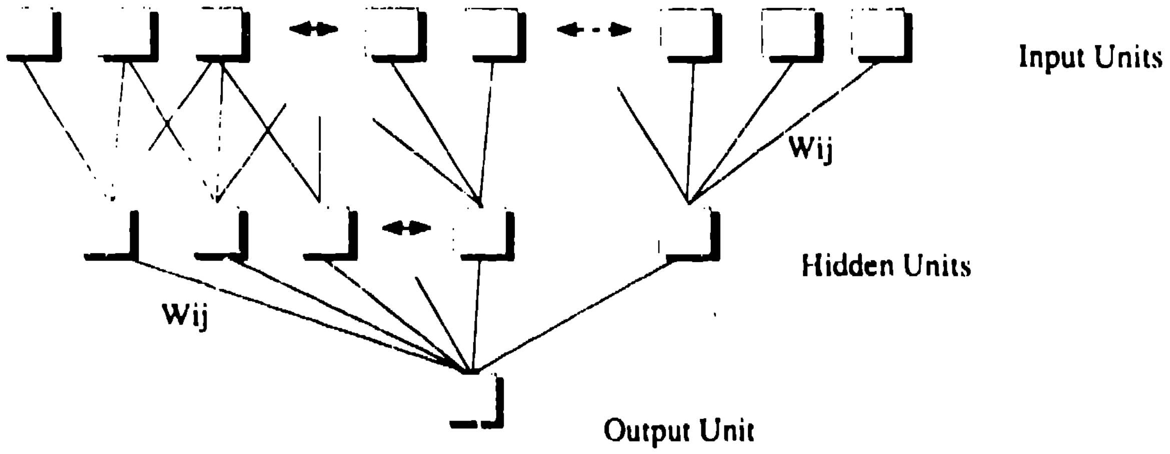
(A) E. coli Promoters



(B) Perceptron



(C) Back Propagation


Input Units
Wij
Hidden Units
Wij
Output Unit